

Uwaga: we wszystkich programach należy założyć, że dołączone są biblioteki iostream, stdlib oraz dostępna jest przestrzeń nazw std. Sprawdzaniu podlegają jedynie miejsca wyznaczone na odpowiedź. W przypadku stwierdzenia błędu lub niejednoznaczności w pytaniu, należy czytelnie napisać komentarz wyjaśniający napotkany problem. Test oceniany jest w skali 0-100 pkt (próg zaliczenia = 50%).

Zad. 1. (20 pkt. = 2*10 pkt.)

Jaką wartość zwróci poniższe wywołanie funkcji *f* dla następujących parametrów?

```
char x1[] = { "bit" };
char x2[] = { "program" };
```

Odpowiedź:

f(x1) = **10**

f(x2) = **20**

```
int f( char *s ) {
    int z = 0;
    while ( *(s+z) != '\0' )
        z++;
    *(s+z/2) = '\0';
    if ( z <= 1 )
        return z+4;
    else
        return f( s ) + f( s + z/2+1 );
}
```

Zad. 2. (21 pkt. = 3*7 pkt.) Podaj zawartość struktury *s* bezpośrednio przed zakończeniem realizacji funkcji *main*.

Odpowiedź:

s.x = **1**

s.y = **0**

s.z = **2**

```
typedef struct {
    int x, y, z;
} type_t;
void f( type_t x ) { x.y = 3; }
void g( type_t x[] ) { x->x = 1; f( *x ); }
void h( type_t **x ) { (*x)->z = 2; }
int main() {
    type_t s = { 0, 0, 0 }, *ps = &s;
    f(s);
    g( &s );
    h( &ps );
    return 0;
}
```

Zad. 3. (19 pkt.) Podaj co pojawi się na wyjściu programu w wyniku wykonania podanego programu.

Odpowiedź: **margorp**

```
#define SIZE 12
int main() {
    char s[SIZE], str[] = { "programming" };
    int i=0, j=6;

    while ( str[i] != '\0' ) {
        s[j--] = str[i++];
        if ( j < 0 )
            j = SIZE - 1;
    }
    s[j] = str[i];
    cout << s;
    return 0;
}
```

Zad. 4. (20 pkt. = 5*4 pkt.) Podaj tekst, który zostanie wypisany na wyjściu w wyniku wykonania poszczególnych instrukcji "cout" (w miejscu na odpowiedź oznaczony etykietą "Instrukcja x" wpisz tekst wypisany przez instrukcję "cout" z komentarzem /* I-x */). Wpisz ERR jeśli nie można jednoznacznie stwierdzić co zostanie wypisane na ekran. Kodowanie liczb w systemie binarnym przyjmujemy tak jak omówiono na wykładzie, tzn. U2. Jeśli dana instrukcja powoduje błąd wykonania programu, to w odpowiedzi wpisz ERR i kontynuuj realizację programu z pominięciem tej instrukcji.

Odpowiedzi:

Instrukcja 1: **22**

Instrukcja 2: **ERR**

Instrukcja 3: **-5**

Instrukcja 4: **ERR**

Instrukcja 5: **3**

Zad. 5. (20 pkt. = 5*4 pkt.)

Wyróżnione pola uzupełnij, tak aby program poprawnie się komplikował, bezbłędnie wykonywał oraz wypisywał na ekran liczby:

8

Alokacje pamięci powinny być wykonane w taki sposób, aby każde wywołanie funkcji *malloc* zwróciło adres do najmniejszej możliwej tablicy skutkującej poprawnym wykonaniem programu. Zakładamy, że podczas realizacji programu, każde wywołanie funkcji *malloc* zwraca wskaźnik różny od *NULL*.

```
#define SIZE 6

int * f( int tab[], int z ) {
    int *x = (int *) malloc( z*sizeof(int) );
    int i, j=0;
    for ( i=0; i < z; i++ )
        if ( tab[i] < 0 )
            x[j++] = tab[i];
    while ( j < z )
        x[j++] = 0;
    return x;
}

int main() {
    int w[] = { -2, 3, 5, -1, 4, -3, 2, -4 }, *z[2], d, i;
    z[0] = f( w, 2 );
    d = z[0][0]+z[0][1];
    z[1] = f( w, SIZE );
    for ( i=0; i < SIZE; i++ )
        d += z[1][i];
    cout << -d;
    free( z[0] ); free( z[1] );
    return 0;
}
```